Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Matic Likozar

Prepoznava oseb na osnovi očesne šarenice

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: viš. pred. dr. Borut Batagelj

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Kandidat naj pregleda področje sistemov za prepoznavanje oseb na podlagi šarenice. Opiše naj Daugmanovo metodo, ki je najbolj razširjena in jo primerja z drugimi sistemi. Na osnovi znanja različnih sistemov naj implementira sistem za prepoznavo oseb na osnovi šarenice in predstavi in komentira svoje rezultate.

Kazalo

Seznam uporabljenih kratic

kratica	slovensko
3CCD	kamera, ki zajema vsako barvno komponento posebej
DNA	deoksiribonukleinska kislina
XOR	ekskluzivni ali, ekskluzivna disjunkcija
ICA	analiza neodvisnih komponent
1D	eno dimenzionalno
2D	dvo dimenzionalno
3D	tri dimenzionalno
FAR	stopnja napačne potrditve
FRR	stopnja napačne zavrnitve
POS	stopnja pravilnih rezultatov

Povzetek

Naslov: Prepoznava oseb na osnovi očesne šarenice

Biometrija je proces zbiranja, proučevanja in primerjanja posameznikovih fizičnih lastnosti z namenom identificiranja in avtentikacije. Biometrični sistemi omogočajo avtomatsko identificiranje posameznika na podlagi unikatnega vzorca. Identifikacija na podlagi očesne šarenice predstavlja enega najbolj zanesljivih sistemov identificiranja posameznikov. Večina komercialnih sistemov deluje na osnovi Daugmanove metode, katera je pokazala odlične rezultate, tako v časovni zahtevnosti algoritma, kot tudi v stopnji uspeha prepoznavanja oseb.

Namen dela je predstaviti osnovni postopek delovanja sistema za prepoznavanje očesne šarenice. Sistem obsega pet glavnih metod: zajem slike šarenice, segmentacija šarenice, normalizacija, kodiranje vzorca in primerjanje vzorcev. Podrobno bo predstavljena vsaka metoda Daugmanovega sistema in za primerjavo še Wildov sistem. Na podlagi predstavljenih metod bomo na koncu predstavil še svojo rešitev za prepoznavanje šarenice.

Ključne besede: biometrija, prepoznavanje šarenice, segmentacija, normalizacija, vzorec šarenice .

Abstract

Title: Iris Recognition for Person Identification

Biometrics is the process of collecting, studying and comparing the individual's physical characteristics in order to identify and authenticate. Biometric systems allow automated identification of an individual based on the unique pattern. Identification based on iris represents one of the most reliable systems for identifying individuals. Most commercial systems operate based on Daugman's method, which showed excellent results, both in the time complexity, as well as the rate of success when identifying persons.

The aim of this work is to present the basic operation of the system for identifying the iris. The system comprises five main methods: image acquisition, iris segmentation, iris normalization, feature extraction and pattern matching. Details will be presented by each method of Daugman's system and compared with Wilde's system. Based on presented methods we will develop our own method for iris recognition.

Keywords: biometric identification, iris recognition, segmentation, normalization, iris pattern .

Poglavje 1

Uvod

Biometrija je merjenje in analiza fizičnih in vedenjskih značilnosti ljudi. Beseda "biometrics"je izpeljanka iz grških besed "bio", kar pomeni življenje in "metric", kar pomeni meriti. Tehnologija je pogosto uporabljena za prepoznavanje in kontrolo dostopa ljudi. Vodilna točka biometrije je v tem, da ima vsak človek edinstvene značilnosti in ga je moč prepoznati po njegovih fizičnih ali vedenjskih značilnostih. Biometrija se deli v dve skupini, fizične in vedenjske značilnosti. Fizične značilnosti v biometriji za avtenticiranje so prstni odtis, DNA, obraz, obris roke, očesna mrežnica, oblika uhlja in očesna šarenica, ki bo podobno predstavljena v tem delu. Vedenjske značilnosti so povezane z obliko obnašanja osebe. To so pisava, hoja, kretnje in govor.

Avtentikacija z biometrijo je vse bolj uporabljena v varnostnih sistemih korporacij, javnih varnostnih sistemih, napravah in aplikacijah. Dobra stran biometrije je tudi v tem, da ne potrebujemo gesla, katerega si moramo zapomniti, ampak imamo "geslo"vedno pri sebi. Za branje avtentikacijskih značilnosti potrebujemo napravo, kot je na primer, čitalec prstnega odtisa, programsko opremo, katera obdela odčitane informacije v digitalno obliko in podatkovno bazo, ki hrani biometrične podatke za iskanje ujemanja.

Biometrije ima tako slabe, kot dobre strani. Kvaliteta biometrije se skozi čas izboljšuje, ampak je še vedno možnost, da dobimo lažno negativne (FRR) in lažno pozitivne (FAR) rezultate. Problem pri biometriji je v tem, da se jo

lahko ponaredi. Ljudje puščajo prstne odtise vsepovsod in jih je enostavno kopirati in narediti lažni prstni odtis. Prav tako ljudje puščajo DNA vsepovsod, tudi zvok je moč posneti. Ljudje ne morejo spremeniti biometričnih značilnosti, kot to lahko naredijo z geslom.

V primeru avtentikacije je najbolj primerna uporaba biometrije skupaj z uporabo gesel. Poznamo več stopenj zanesljivosti. Z uporabo biometrije si zagotovimo večjo stopnjo zanesljivosti. Stroški biometrije so se do danes močno zmanjšali, saj je danes naprava za zaznavanje biometričnih značilnosti že kar mobilni telefon. Že večina telefonov ima danes kvalitetno kamero, nekateri pa imajo tudi čitalec prstnih odtisov. S tem je biometrijo vedno lažje pripeljati v javno uporabo za avtentikacijo ljudi.

1.1 Razvoj biometrije

Biometrija sega v leto 1858, ko so prvič zajeli sliko odtisa roke z namenom identifikacije oseb. Kasneje leta 1892 je bil razvit prvi sistem za branje prstnega odtisa, katerega so sprva uporabljali v New Yorškem zaporu. Leta 1936 je bil predstavljen koncept sistema za branje očesne šarenice, kateri je bil dokončno razvit in patentiran leta 1994. Med tem so se do dobra razvili sistemi za branje prstnih odtisov, odtisov dlani in predstavljen je bil koncept prepoznavanja obrazov. Leta 1995 je bil izdan prvi prototip sistema, za branje šarenice, za komercialno uporabo. Med letoma 2000 in 2005 se je na podlagi biometrije razvilo veliko sistemov in aplikacij. Leta 2005 je bil predstavljen koncept sistema za branje šarenice na daljavo. Leta 2010 v Združenih državah uvedejo biometrijo za boj proti terorizmu. Leto kasneje preteče patent, katerega je implementiral John Daugman. Leta 2013 je prvič uporabljena biometrija v mobilnih napravah. Sistem za branje šarenice je bil prvič uporabljen v mobilnih napravah leta 2016.

1.2 Prepoznavanje na podlagi šarenice

Prepoznavanje oseb na podlagi šarenice ni nova ideja, vendar pa je v praksi dostopna zadnjih 15 let. Kot varnostni sistem metoda ni pogosto uporabljena zaradi visoke cene. Sistem prepoznavanja šarenice bo v prihodnosti eden izmed najbolj zanesljivih varnostnih sistemov. Sistem je zanesljiv prav zaradi unikatnosti šarenice. Unikatnost šarenice se pokaže že pri eni osebi. Levo in desno oko imata različno teksturo šarenice. Enako je pri dvojčkih. Pri testiranju DNA, se pokaže, da je pri dvojčkih rezultat enak. Pri šarenici pa imamo pri primerjanju dvojčkov 4 unikatne vzorce šarenic [6]. Zaradi unikatnosti in težjega ponarejanja vzorca se ti sistemi vedno bolje uporabljajo. Zaradi visoke zanesljivosti in hitrega delovanja je uporaba teh sistemov na letališčih in drugih javnih mestih vedno bolj pogosta. Kot je bilo dokazano, ima to vrstna biometrija preko 90 odstotno stopnjo zanesljivosti pri prepoznavanju in avtenticiranju osebe.

1.3 Pregled znanih metod prepoznavanja šarenice

Skozi leta proučevanja uporabe šarenice v biometriji se je razvilo več vrst metod za prepoznavanje oseb na podlagi šarenice.

Signalno usmerjene metode

Signalno usmerjene metode [5][11] prepoznajo vzorec šarenice na podlagi informacije signalov. Informacija signala je neodvisna od barvne intenzitete in svetlosti. John Daugman je prvi implementiral in predstavil komercialni sistem na podlagi signalov leta 1994. Kodni vzorec je binarni zapis ustvarjen z uporabo signalov, kateri so rezultat teksturnih filtrov. Poleg vzorca je ustvarjena tudi maska, ki pove kateri del vzorca je pokvarjen in ni dober za uporabo. Za primerjavo vzorca je uporabljen operator XOR.

Metode na podlagi teksture

Wild je predstavil metodo za prepoznavanje vzorca šarenice na podlagi teksture [5][11]. Za delovanje je potrebna slika visoke kvalitete. Za lokalizacijo šarenice se uporablja Houghova transformacija, katera poišče dva kroga, ki tvorita notranji in zunanji rob. Za spodnjo in zgornjo veko pa se uporabi krivulji. Za vzorec se uporablja Laplacov filter, iz katerega se sestavi Laplacova piramida. Primerjava se izvaja na podlagi korelacije zajete slike in slike shranjene v podatkovni bazi.

Metode Zero-Cross

Metoda predstavlja vzorec šarenice na različnih resolucijskih nivojev. Vzorci so ustvarjeni na podlagi valovnega "Zero-corssinga" [5][11]. "Zero-cross" je točka, kjer spekter valovanja seka os signala. Iz vhodne slike izračunamo 1D signal in njegovo "zero-cross" vrednost. Segmentacija deluje na podlagi zaznavanja robov in uporabe houghovoe transformacije. Za primerjanje vzorca uporabimo vrednosti "zero-cross" izračunane iz signalov za vsak resolucijski nivo. Dobra stvar pri tej metodi je, da imamo manj računanja saj je število "zero-cross" točk manj kot vseh točk. Slabost pa je v tem, da za primerjanje potrebujemo enako število "zero-cross" točk v vsakem nivoju.

Metode na podlagi barvnih sprememb

Sistem deluje na podlagi lokalnih barvnih sprememb [5][11]. Visoke spremembe barvitosti so zajete pri izračunu vzorca. Parametri šarenice so izračunani z uporabo houghove transformacije Canny edge metode [10] zaznavanja robov na sliki. Za izračun vzorca najprej izračunamo 1D signal intenzitet iz katerega nato izračunamo vektor vzorca. Vrednosti vektorja so povprečne vrednosti odklona magnitude v 8x8 blokih normalizirane šarenice. Podobnost vzorcev se izračun z uporabo operacije XOR. Za klasifikacijo primerjav se uporablja Fisherjeva linearna diskriminanta.

Metode z uporabo neodvisne analize

Metoda za izračun kodnega vzorca uporablja analizo neodvisnih komponent (ICA)[5][11]. Pri tej metodi šarenico normaliziramo tako, da zajamemo fiksno število krogov od notranjega roba šarenice proti zunanjemu roba. Iz teh krogov sestavimo tabelo $N \times M$. S tem je metoda odporna na napake zaradi rotacij in velikosti vhodne slike. Neodvisne komponente so med seboj nepovezane. Komponente so določene iz koeficientov vzorca šarenice. Iz normalizirane šarenice se izračunajo centri za vsak razred z uporabo "konkurenčnega učenja"in so shranjeni kot kodni zapis vzorca.

Metode dinamičnega programiranja

metoda za avtenticiranje uporablja kinematične lastnosti šarenice [5][11]. Za lociranje zenice se najprej poišče najvišjo intenziteto na sliki. Na podlagi te vrednosti na sliki označimo vse povezane elemente ki imajo intenziteto manjšo od maksimalne. Z izborom maksimalne vrednosti dobimo center zenice. Z uporabo kvadrata obrežemo šarenico. Za nadaljno primerjanje karakteristik dobljenih oblik se prav tako uporabi dinamično programiranje. Za primerjanje dveh šarenic se upošteva minimalne razdalje med podobnimi oblikami.

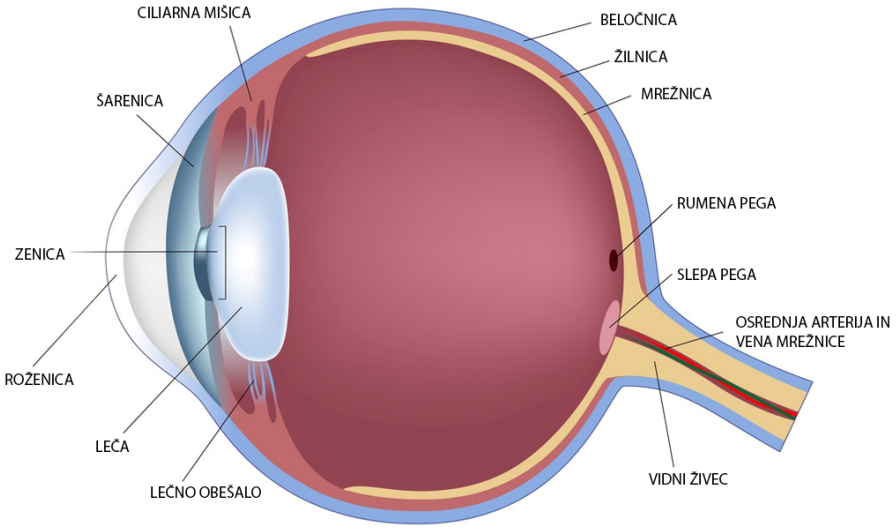
Poglavje 2

Sestava očesa

Oko je čutilo, ki zaznava svetlobo. Sestavljeno je iz približno 2,5 cm debelega kroglastega očesnega zrkla, ki je vsajeno v koščeno ležišče, imenovano očesna votlina ali očnica. Zrklo obdajajo tri plasti ovojnic. Zunanja je čvrsta beločnica z zdrsljivo površino, ki sestavlja belino očesa. Osrednja plast je žilnica, ki oskrbuje oko s krvjo. Arterija in vena mrežnice dobavljata kri žilnici. Notranja plast je mrežnica, v kateri so za svetlobo občutljive celice. Zrklo je s prožno lečo predeljeno v dva neenaka prostora. Vid nam omogočajo oči, ki zbirajo in usmerjajo svetlobno energijo, prihajajočo s predmetov in jo spremenijo v sliko. Slika predmeta pade na celice, imenovane fotoreceptorji, ki potem pošiljajo signale v možgane. Na ta način prihaja v možgane velikanska količina informacij.

Leča očesa (slika 2.1) je s suspenzornimi ligamenti oziroma lečnim obešalom pritrjena na cilarično mišico (slika 2.1). Mišica se krči in širi in s tem spreminja obliko leče. Pri gledanju na daleč je ciliarna mišica sproščena, tako da so vlakna, ki jo povezujejo z lečo, napeta in se leča splošči. Kadar se ciliarna mišica skrči, se napetost vlaken zmanjša, zato se leča vrne v bolj izbočeno in okroglo obliko. Pri človeku se s staranjem tovrstna prožnost zmanjša, zato se oko ni več sposobno osredotočiti na bližnje predmete.

Rumena pega (slika 2.1) je odgovorna za oster sredinjski vid, ki je za človeka pomemben pri aktivnostih, kot so branje, pisanje in vožnja, za katere



Slika 2.1: Oko

so vidni detajli pomembni. Poleg rumene pega, ima oko tudi slepo pego (slika 2.1). Slepa pega je majhno in za svetlobo ne občutljivo območje mrežnice. Na tem mestu izhaja iz zrkla vidni živec (slika 2.1), zato tu ni čutnic za zaznavanje svetlobe. Po navadi se slepe pege ne zavedamo, ker možgani pri izdelavi slike "ignorirajo" praznino, ki se pojavlja zaradi ne občutljivosti slepe pege.

S pomočjo očesa se zunanji dražljaji preko vidnega živčevja prenašajo v možgane, ki te impulze pretvorijo v sliko. Slika, ki nastane v možganih, je v bistvu naše vidno zaznavanje.

Vsako očesno zrklo ima lečo iz prozorne beljakovine. Obročaste ciliarne mišice spreminjajo obliko leče in s tem omogočajo izostrovanje slike predmetov. Temu procesu pravimo tudi akomodacija. Leča vrže na mrežnico (slika 2.1) obrnjeno sliko predmeta. Možgani obrnejo sliko, zato se sploh ne zavedamo, da so na mrežnici slike vseh predmetov postavljene na glavo. V starosti postane očesna leča bolj toga, zaradi česar nastane daljnovidnost.

Roženica (slika 2.1) ščiti prednji del očesa in pomaga pri usmerjanju svetlobnih žarkov, kateri prihajajo v oko. Pokriva jo mrena, imenovana veznica, ki je tudi na notranji površini vek. Veznico nenehno vlažijo solze, ki so ne-

kakšno mazivo za očesno zrklo. V solzah je lizocim, snov, ki ubija bakterije in preprečuje okužbe.

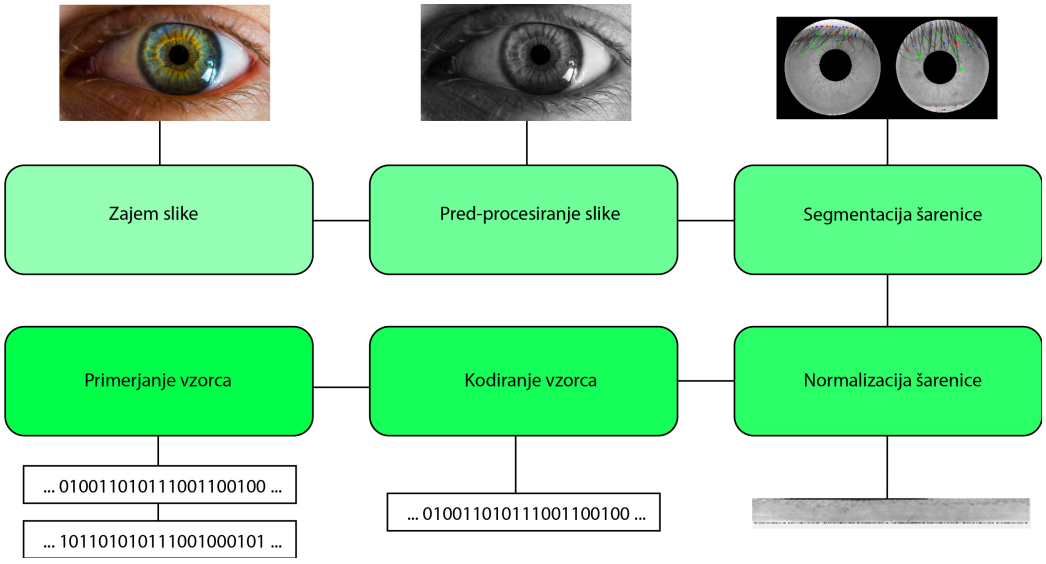
Zenica (slika 2.1) je odprtina skozi katero v oko prihajajo svetlobni žarki. Šarenica (slika 2.1) je kolobar mišičnega tkiva med roženico in lečo in spreminja velikost zenice. Pri premočni svetlobi šarenica zenico zoži, pri šibki pa jo razširi. V šarenici je gladka mišica, ki deluje refleksno. Mišično tkivo ima barvila, katera dajejo očem barvo (slika 2.2).

Slika 2.2: Šarenica

Poglavje 3

Prepoznavanje šarenice

Prepoznavanje šarenice je biometrična metoda avtenticiranja oseb na podlagi mišičnega tkiva v očesu. Slika 3.1 prikazuje postopek delovanja sistema za prepoznavanje šarenice.



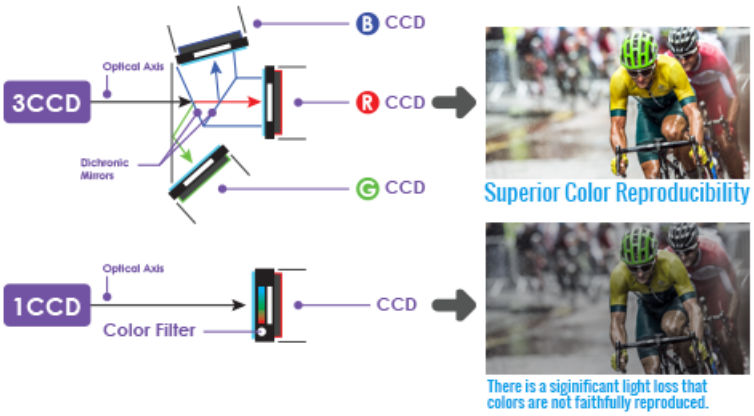
Slika 3.1: Delovanje sistema za prepoznavanje šarenice

Šarenico je mogoče zajeti v kamero, katera je oddaljena od 10 cm do nekaj metrov. Za zajem unikatnega vzorca je potrebno zajeto sliko obdelati. Vzorec nato pretvorimo v bitni zapis in tako dobimo biometrično informacijo, katero lahko primerjamo z ostalimi vzorci v podatkovni bazi. Za celoten

postopek prepoznavanja šarenice, je potrebnih šest korakov. Prvo potrebujemo kvalitetno sliko šarenice, temu postopku pravimo zajemanje slike. V postopku predprocesiranja slike, moramo dobljeno sliko pretvoriti v sivinsko sliko. V tem postopku odstranimo tudi šume, ki so nastali ob zajemanju slike. Ko imamo sliko pripravljeno za obdelavo, moramo iz slike odstraniti vse nepotrebne dele. Temu postopku pravimo segmentacija slike. Segmentiranje slike nam vrne kolobar, katerega moramo nato normalizirati v pravokotnik. Postopku pretvarjanja kolobarja v kvadrat pravimo normalizacija. Iz normalizirane oblike šarenice nato izračunamo kodni zapis s katerim primerjamo zapise v podatkovni bazi.

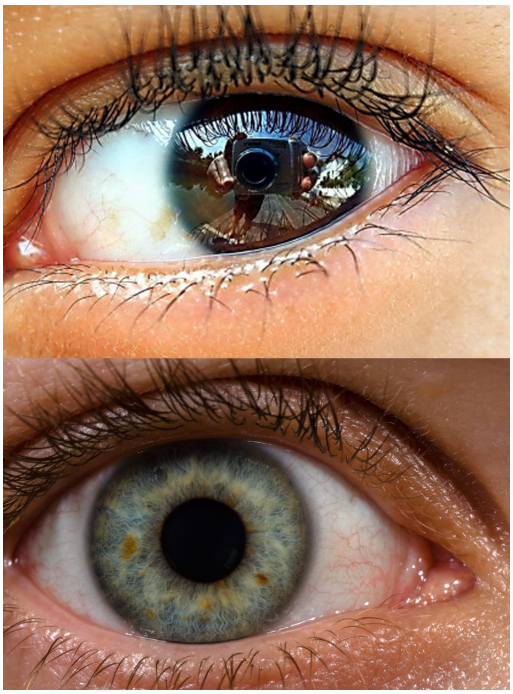
3.1 Zajemanje slike očesa

Zajemanje slike očesa je prvi korak v postopku prepoznavanja šarenice. Ta korak je tudi najbolj pomemben, saj so vsi ostali koraki odvisni od kvalitete slike. Pomembno je, da pridobimo čimbolj kvalitetno sliko. Kvalitetna slika pripomore k lažjemu odstranjevanju šumov in preprečuje napake pri izračunavanju. Za zajemanje slike očesa je najbolj primerna 3CCD kamera. 3CCD kamera omogoča zajem boljše kvalitete barv na sliki 3.2, saj zajema vsako komponento barve posebej.



Slika 3.2: 3ccd kamera

Prepoznavanje šarenice deluje na principu barvnosti očesne šarenice, zato kvalitetna barva pripomore k izračunavanju vzorcev. Poleg barve pa je pomembna tudi osvetlitev očesa ob zajemanju slike. Paziti moramo, da odsevi svetlobe ne prekrivajo šarenice na sliki 3.3, saj s tem izgubimo pomembne informacije.

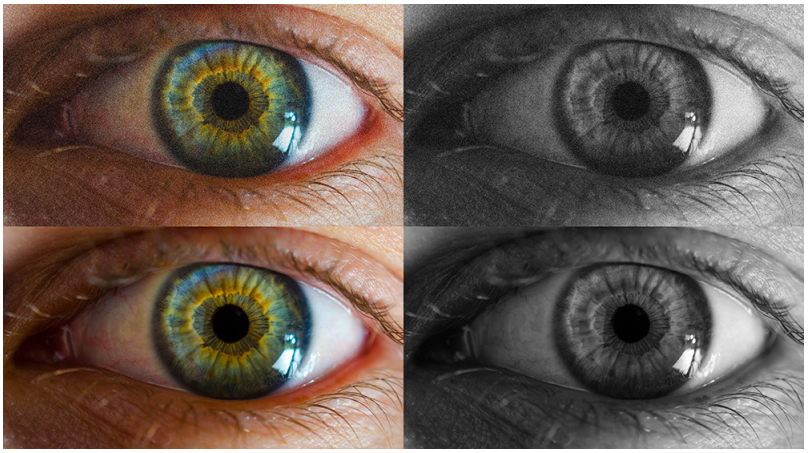


Slika 3.3: Primer dobre (spodaj) in slabe (zgoraj) slike očesa, kjer je viden svetlobni odsev

3.2 Predprocesiranje slike

V postopku predprocesiranja pripravimo sliko za izračun biometričnega vzorca. V tem postopku popravimo popačenje slike. Do popačenja slike privede več dejavnikov. Eden od dejavnikov je postavitev kamere, kateri privede do popačenja očesa. Popačenost slike dobimo tudi, če je drža obraza med zajemanjem slike napačna. Prav tako v tem postopku odstranimo oziroma zmanjšamo prisotnost šuma, ki je nastal med zajemanjem slike 3.4. Pred-

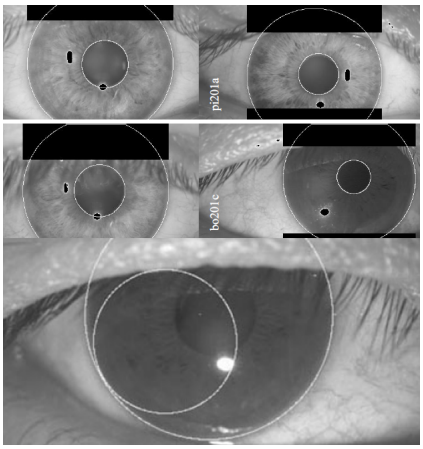
procesiranje privede popravljeno sliko do te mere, da je čimbolj podobna originalni in da ustreza potrebam za prepoznavanje šarenice. Glavni namen pred-procesiranja slike je pretvorba slike v sivinsko sliko. Tako 24 bitno sliko pretvorimo v 8 bitno in s tem omogočimo lažje procesiranje in zmanjšamo prostorsko zahtevnost.



Slika 3.4: Šum in črno-bela slika

3.3 Segmentacija očesa

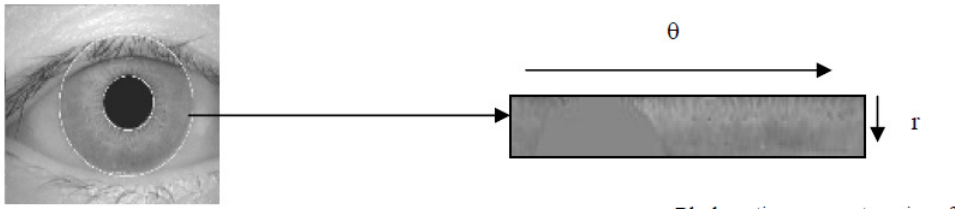
Postopek segmentacije iz slike očesa loči kolobar šarenice. Šarenico ločimo z dvema krogoma, prvega postavimo na rob med šarenico in beločnico, drugega pa na rob med šarenico in zenico. Tako iz slike celotnega očesa dobimo le kolobar šarenice. Kolobar lahko vsebuje trepalnice in svetlobne odseve, katere moramo prav tako odstraniti. Segmentacija je kritična točka celotnega postopka prepoznavanja šarenice, saj lahko dobimo napačen vzorec. Največji problem pri ločitvi šarenice je pigment šarenice. Če je pigment temen, bo težje zaznati rob med šarenico in zenico, kar pa lahko privede do napačnega končnega rezultata kot je prikazano na sliki 3.5.



Slika 3.5: Segmentacija slike očesa

3.4 Normalizacija šarenice

Ko imamo kolobar šarenice ločen, ga moramo v koraku normalizacije pretvoriti v obliko s fiksnimi dimenzijami. Preoblikovanje kolobarja v pravokotnik, dosežemo s preslikovanjem polarnih točk v kartezične točke. Proces normalizacije omogoči, da ima območje šarenice konstante dimenzije in iste lastnosti na neki točki. Kar pomeni, da dve sliki iste šarenice zajete pod različnimi pogoji vrneta vzorca, katera imata iste lastnosti na točki $T(xn, yn)$. Na sliki 3.6 je prikazano, kako izgleda normalizirana šarenica.



Slika 3.6: Normalizacija šarenice

3.5 Kodiranje vzorca

Da dobimo natančno prepoznavanje oseb, moramo iz normalizirane šarenice vzeti le informacije, ki so pomembne, dojemljive in unikatne. Iz teh informacij nato izračunamo unikaten vzorec. Vzorec je bitni zapis številk in ustrezno masko šumov, katera določa dobre in pokvarjene dele vzorca. Dobljeni vzorec mora nato ustrezati ujemanju enakosti dveh vzorcev šarenice. Prvi vzorec se izračuna, ko se uporabnik registrira, drugi vzorec pa se izračuna, ko se uporabnik avtenticira. Kodiranje dveh enaki šarenic in kodiranje različnih šarenic mora vrniti različen rezultat. Vzorci enakih šarenic morajo biti zelo podobni in vzorci različnih šarenic morajo biti zelo raznoliki, da lahko postavimo trdno odločitev, ko bomo primerjali vzorce.

3.6 Primerjava in ujemanje vzorca

Primerjanje vzorcev poteka na podlagi pridobljenih vzorcev. Vzorec, ki smo ga pridobili v času registracije uporabnika primerjamo z vzorcem, ki smo ga dobili v času avtenticiranja. Vzorca primerjamo tako, da izračunamo razliko bitnih zapisov. Pri primerjanju moramo upoštevati tudi masko šumov. Primerjamo le tiste bite, ki so v maski šumov označeni kot ne-pokvarjeni. Razliko bitnih zapisov dobimo tako, da štejemo razlike med vzorci in shranjujemo rezultate. Rezultat, ki bo vrnil najmanjšo odstopanje vzorcev, bo veljaj za končno odločitev. Na podlagi končnega rezultata se moramo nato odločiti ali gre za vzorca iste šarenice.

Poglavje 4

Opis metod za prepoznavanje šarenice

4.1 Metode za segmentacijo šarenice

Za učinkovito prepoznavanje očesne šarenice je segmentacija zelo pomemben postopek. Poznamo več metod za segmentiranje. Najbolj poznani metodi sta Wildova [4] in Daugmanova [1] metoda segmentiranja očesne šarenice.

Wilde je predstavil metodo segmentiranja v dveh korakih. Prvi korak zajema izgradnjo "robnega zemljevida", kar pomeni, da na sliki poiščemo robove. Robove je moč zaznati s pomočjo intenzitet pikslov na sliki očesa. V drugem koraku poiščemo zunanji rob šarenice in notranji rob šarenice. To naredimo z uporabo Houghove transformacije.

Daugmanova metoda deluje na osnovi integralno diferencialne operacije, katera na sliki očesa poišče rob med beločnico in šarenico ter rob med šarenico in zenico. Metoda poišče robove tam, kjer je največja razlika med intenzitetami pikslov na sliki.

4.1.1 Wildova metoda segmentiranja

Richard Wilde je leta 1996 predstavil svojo metodo avtomatskega segmentiranja očesne šarenice na podlagi Houghove transformacije in zaznavanja

robov na sliki. Prvi korak metode je zaznavanje robov.

Zaznavanje robov na sliki

Za zaznavanje robov na sliki je uporabljen Cannyjev algoritem detekcije robv [10]. Canny detektor robov je algoritem za zaznavanje robov. Uporablja večstopenjski algoritem za odkrivanje robov na slikah. Razvil ga je John F. Canny leta 1986.

Prvi korak pri zaznavanju robov na sliki je zmanjševanje šuma slike. To naredimo z Gaussinovim filtrom. Običajno uporabimo matriko 5×5 s katero se "sprehodimo" po sliki in pomnožimo vrednosti pikslov in s tem zgladimo sliko.

Iz zglajene slike nato naredimo dve novi sliki, tako da ju filtriramo s horizontalnim in vertikalnim 3×3 Sobel operatorjem. Če je A naša glajena slika in jo filtriramo s horizontalnim filtrom dobimo magnitudo G_x oziroma, če jo filtriramo z vertikalnim filtrom dobimo magnitudo G_y.

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * A$$

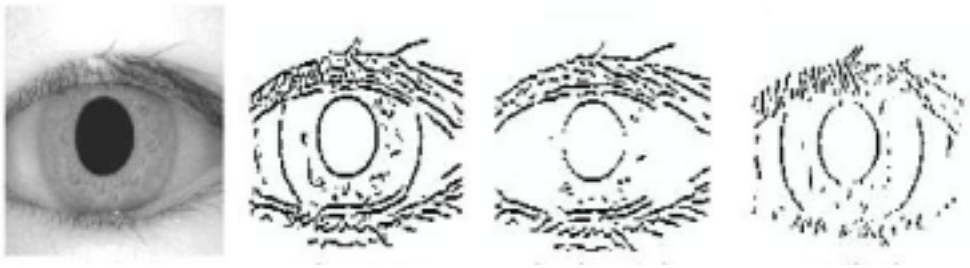
Nato iz soležnih točk iz slike G_x in G_y izračunamo robno magnitudo G $G = \sqrt{G_y^2 + G_y^2}$ in za vsako magnitudo še smer po formuli $s(\theta) = \tan^{-1}(\frac{G_x}{G_z})$.

Ko imamo izračunane magnitude in njihove smeri, znova pregledamo vsak piksel in primerjamo njegovo magnitudo G s sosednjimi v smeri $s(\theta)$. Če je magnituda lokalni maksimum, piksel na tej lokaciji obdržimo, drugače pa mu nastavimo vrednost na 0. Tako obdržimo le piksle z največjimi vrednostmi magnitud.

V zadnjem koraku iz obdržanih pikslov določimo tiste, kateri tvorijo dejanske robove na sliki. Da določimo kateri piksli tvorijo robeve, potrebujemo dve konstatni P_{max} in P_{min}. Če se vrednost magnitude piksla nahaja nad

P_{max} vrednostjo, piksel tvori dejanski rob. Če je vrednost magnitude pod P_{min}, piksel ne tvori roba na sliki. Za vse vrednosti, ki se nahajajo vmes, pa je potrebno pregledati sosednje piksle. Če obstaja sosednji piksel, ki tvori rob, potem se tudi iskani piksel določi kot del roba. S tem postopkom odstranimo tudi vse manjše piksle, kateri tvorijo nekakšen šum.

Tako dobimo sliko, katera tvori močne robove na sliki 4.1.



Slika 4.1: Zaznavanje robov

Houghova transformacija

Houghova transformacija je metoda prepoznavanja preprostih oblik na slikah. Metodo je patentiral Paul Hough leta 1962, dokončno pa sta jo razvila Richard Duda in Peter Hart leta 1972.

Da poiščemo krog na sliki, potrebujemo Matriko robov iz prvega koraka. Nato na tej sliki izvedemo Houghovo transformacijo. Ker iščemo kroge na sliki, potrebujemo tri dimenzionalno tabelo $H(X_0, Y_0, r)$. Tabelo zgradimo glede na parametre slike. X in Y sta višina in širina slike, r pa predstavlja največji možni polmer kroga $r = min(\frac{X}{2}; \frac{Y}{2})$.

Ko imamo 3D tabelo in vse njene vrednosti nastavljene na 0, se "sprehodimo" čez sliko in za vsako točko (X,Y) na sliki preverimo, če se nahaja na območju kjer je rob. Če je na tej točki rob, potem se ponovno "sprehodimo" čez celotno sliko in za vsako točko (X_0,Y_0) in vsako možno vrednost r preverimo če se krog dotika roba. To preverimo tako, da izračunamo vrednost $G(X, Y, X_0, Y_0, r) = (X - X_0)^2 + (Y - Y_0)^2 - r^2$. Če je vrednost G na točki

(X_0,Y_0) enaka 0, v naši 3D tabeli na poziciji (X_0, Y_0, r) povečamo vrednost za 1.

Ko na ta način "prehodimo" celotno sliko in imamo 3D tabelo napolnjeno z vrednostmi, poiščemo polje v tabeli, ki ima največjo vrednost. Polje z največjo vrednost tvori središče našega iskanega kroga na sliki. Prav tako vemo dolžino polmera za najdeni krog.

Ko imamo zunanji in notranji rob šarenice, je potrebno poiskati še rob, kjer šarenico prekrivata veki. To naredimo tako, da na območju šarenice določimo spodnji in zgornji del, kjer bomo iskali rob. Območje določimo tako, da vzamemo premer zenice in višino med notranjim in zunanjim robom šarenice. Tako dobimo kvadratno območje, ker bomo iskali rob veke.

Za to, da poiščemo rob med veko in šarenico uporabimo linearno Houghovo transformacijo. Metoda poišče ravne črte na sliki in deluje podobno kot iskanje krogov. Sestavimo tabelo $H(\varphi, \alpha)$ kjer α predstavlja vse možne kote med ordinato X in pravokotnico na črto, ki poteka skozi našo trenutno točko (X, Y) in φ predstavlja dolžino pravokotnice na črto, ki poteka skozi to točko.

Zopet nastavimo vrednosti v tabeli na 0 in se "sprehodimo" čez celotno sliko. Na vsaki točki, katera se nahaja na območju roba, za vsako vrednost α izračunamo vrednost $G(X, Y, \varphi, \alpha) = (X\cos(\alpha) + Y\sin(\alpha)) - \varphi$. Vrednost polja ($\varphi$,$\alpha$) v tabeli H povečamo za 1, če je vrednost G enaka 0. Ko se "sprehodimo" čez celotno sliko, poiščemo v tabeli H maksimalno vrednost in tako dobimo točko, kjer bo potekal rob med veko in šarenico.

Izboljšava delovanja Houghove transformacije

Delovanje Houghove transformacije lahko dosežemo na zelo preprost način. Namesto, da uporabimo združeno robno magnitudo, iz postopka zaznavanja robov, uporabimo le posamezni magnitudi. Za zaznavanje roba šarenice uporabimo le vertikalno magnitudo, saj je na njej veliko manj zaznanih robov, kateri so posledica trepalnic. Za zaznavanje roba med vekama in šarenico uporabimo horizontalno magnitudo.

Uporaba ločenih magnitud doprinese k boljši učinkovitosti zaradi manjšega števila robov, katere je potrebno pregledati.

4.1.2 Daugmanova metoda segmentiranja

Daugman za iskanje robov šarenice in tudi za iskanje spodnje in zgornje meje med vekami uporabi integralno diferencialni operator (integro-differential operator), s katerim preiščemo celotno sliko za največjo vrednost. Integro-diferencialni operator je definiran z naslednjo funkcijo $M_{r,x,y} = |G_\theta(r) \oplus \frac{\alpha}{\alpha r}\int_{r,x_0,y_0}(\frac{I(x,y)}{2\pi r}ds)|$, kjer je $I(x,y)$ zajeta slika, G_θ predstavlja funkcij glajenja, ds predstavlja krožni lok iskanja, $\oplus$ pa predstavlja operacijo konvolucije, ki je v matematiki postopek, ko iz dveh funkcij (f in g) dobimo tretjo funkcijo.

Sprva nastavimo funkcijo glajenja na visok faktor, in s tem omogočimo lažje iskanje roba med beločnico in šarenico, saj je tam razlika v intenzitetah pikslov zelo visoka. Ko poiščemo največjo vrednostno razliko, se prične ponovno iskanje na sliki, z namenom, da najdemo še rob med šarenico in zenico. Pri tem uporabimo bolj "fino"funkcijo glajenja, saj je običajno intenzitetna razlika roba šarenice in zenice zelo nizka. Tako v prvem, kot tudi v drugem krogu iskanja robov, iščemo le v desnem in levem krožnem loku s kotom 90°, saj vemo, da spodnji in zgornji lok ponavadi prekriva veka. Za vsako točko na sliki izračunamo vrednost za vse možne vrednosti r.

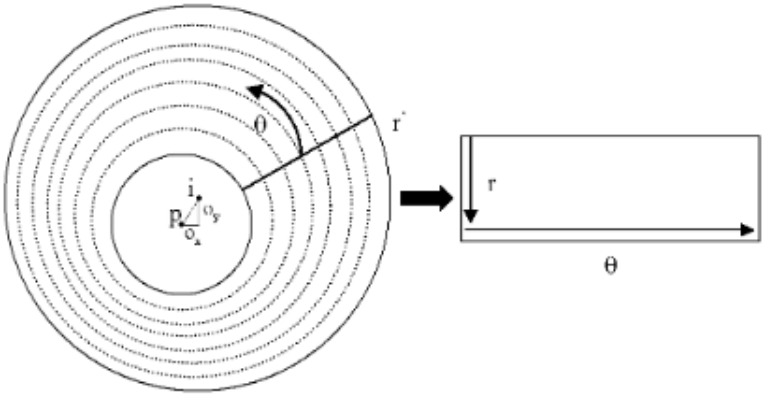
4.2 Metoda za normalizacijo šarenice

V postopku normalizacije, območje šarenice spremenimo do te mere, da ima enake dimenzije pri primerjanju dveh slik iste šarenice. Do nekonsistentnosti šarenice privede širjenje in krčenje zenice pod različno svetlobo, rotacija kamere, nagnjenost obraza, oddaljenost kamere pri zajemanju slike. Dobra metoda normalizacije, mora ustvariti dve različni šarenici, zajeti pod istimi pogoji povsem razlikujoči in dve enaki šarenici, zajeti pod različnimi pogoji v konsistentno obliko. Za normaliziranje šarenice se uporablja Daugmanova

metoda, imenovana "Daugman rubber sheet", prikazana na sliki 4.2.

4.2.1 Daugman rubber sheet

Daugman rubber sheet je linearna metoda, ki za vsako polarno točko neodvisno od dimenzije ustvari kartezično točko določenih dimenzij. Preslikava polarnih točk $P(r, \theta)$ v kartezične $T(X, Y)$ je predstavljena s $T(X(r, \theta), Y(r, \theta))$ $\rightarrow P(r, \theta)$, kjer sta $X(r, \theta)$ in $Y(r, \theta)$ predstavljena kot linearna kombinacija zunanjega in notranjega roba šarenice. $X(r, \theta) = (1 - r)X_p(\theta) + rX_s(\theta)$ $Y(r, \theta) = (1 - r)Y_p(\theta) + rY_s(\theta)$ $X_p(\theta)$ predstavlja točko notranjega roba oziroma meje med šarenico in zenico ter $X_s(\theta)$ predstavlja točko zunanjega roba med šarenico in beločnico v smeri središčnega kota θ.



Slika 4.2: Daugman rubber sheet

Ta metoda omogoča odpravljanje nekonsistentnosti, kot so velikost zenice, velikost slike, oddaljenost kamere. Ne more pa odpraviti nekonsistentnosti, ki so nastale zaradi rotacije kamere ali nagnjenosti obraza. To odpravimo oziroma minimiziramo s primerjanjem vzorcev v premikanju (shifting) v smeri θ.

4.3 Metoda kodiranja vzorca

Za Kodiranje vzorca se uporabljajo tako imenovani Gabor filtri. Gabor filtri so v računalniškem vidu uporabljeni za zajemanje informacij iz slik, analiziranje tekstur in ocenjevanje razlik na sliki. Rezultat linearnega Gabor filtra je opredeljen s sinusnim valovanjem pomnoženim z Gaussinovo funkcijo. V prostorskem pomenu se uporabljajo 2D Gabor filtri. Rezultat pri prostorskem zajemu je modulacija Gaussin-kernel funkcije s ploščinsko sinusnim signalom. Za učinkovito zajemanje koherentnih in nekoherentnih teksturnih značilnosti iz slike šarenice, se uporabljajo 2D Gabor filtri.

4.3.1 2D Gabor filter

2D filtre za zajemanje teksturnih značilnosti šarenice je predlagal Daugman leta 1980 kot metodo za analiziranje slik. Idejo je leta 1985 dokončno razvil Denis Gabor, po katerem se tudi imenuje. Dokazal je, da z uporabo 2D filtra pridobimo največjo možno ločljivost med prostorsko frekvenco ("kaj") in orientacijo ("kje") na sliki.

Dvo-dimenzionalni Gabor filter na sliki, $S(x, y)$ je predstavljen z matematično funkcijo:

$$G(x, y) = e^{-\pi[(x-x_0)^2/\alpha^2+(y-y_0)^2\beta^2]} * e^{-2\pi i[(u_0(x-x_0)+v_0(y-y_0)]}$$

Kjer (x_0, y_0) predstavlja pozicijo na sliki (α, β) predstavljata širino in dolžino filtra, (u_0, v_0) predstavlja modulacijo s prostorsko frekvenco $w_0 = \sqrt{u_0^2 + v_0^2}$ in smerjo $\phi_0 = arctan(v_0/u_0)$.

Realni del rezultata je sploščen, da ima volumen enak 0 in s tem dobimo invarianto svetlosti. Za vsak bit rezultata realnega in imaginarnega dela slike s pomočjo kvantizacije dodelimo vrednost 0 ali 1 za negativne in pozitivne vrednosti na sliki.

4.4 Metode primerjanja vzorcev

Ko iz normalizirane slike dobimo kodni vzorec, ga je potrebno še primerjati z že obstoječim vzorcem v bazi podatkov. Primerjanje dosežemo z izračunom korelacije med dvema vzorcema. V teoriji bi moral biti izračun korelacije vzorcev vzetih iz enake šarenice enak 0, ker pa je v praksi možno, da prihaja do pogreškov pri zajemanju vzorcev zaradi že omenjenih nekonsistentnosti, bo rezultat pogosto približek in le redko enak vrednosti 0. Ker smo vzorec v postopku kodiranja kvantizirali, sta možni le dve vrednosti 0 in 1. Metoda primerjanja je zato precej enostavna, saj primerjamo le ali sta si vrednosti na poziciji n enaki in če nista, povečamo vrednost korelacije.

4.4.1 Daugmanova metoda primerjanja

Daugman je za primerjanje uporabil normalizirano Hammingovo razdaljo.

Hammingova razdalja je izračun nekonsistentnosti med dvema enako dolgima nizoma. Prešteje se število znakov v katerih se niza razlikujeta oziroma prešteje se minimalno število substitucij potrebnih, da nek niz pretvorimo v drugega. Na primer Hammingova razdalja med nizom 2**17**3**8**96 in 2**2**3**3**796 je 3.

Ker pa imamo opravka z bitnim zapisom, lahko uporabimo že znano operacijo XOR (ekskluzivni ali). Operacija vrne vrednost 0, če sta oba bita enaka ($XOR_{(0,0)}$ ali $XOR_{(1,1)}$) oziroma vrne vrednost 1, če sta bita različna ($XOR_{(1,0)}$ ali $XOR_{(0,1)}$). S to peracijo enostavno in hitro dobimo korelacijsko vrednost dveh bitov na piziciji n.

Tako dobimo za primer 128-bitnega zapisa A in B sledečo formulo, za izračun procentualnega pogreška nizov $HD = \frac{\Sigma(XOR_{(A_j,B_j)})}{128}$, kjer j predstavlja pozicijo bita med 0 in 127. Časovna odvisnost te metode je enaka j primerjav in je zato precej zanemarljiva. Ker pa tej metodi lahko rečemo "Brute-force" metoda, je lahko ob veliki količini in slabo načrtovani podatkovni strukturi primerjanje vzorcev precej počasno.

Problem rotacije

Kot vemo Daugmanova metoda normalizacije ne omogoča odpravo nekonsistentnosti zaradi rotacije. To lahko odpravimo med postopkom primerjanja z uporabo premikanja vzorca (shifting). Ko računamo Hammingovo razdaljo, enega od vzorcev "shiftamo" in s tem dobimo več rezultatov. Izmed vseh rezultatov nato izberemo tistega, ki ima najnižjo vrednost.

Uporaba B-drevesa

Za hitro primerjanje lahko uporabimo B-drevo, katero omogoča časovno odvisnost $\log(n)$. Drevo hrani bitne zapise urejene po velikosti in je enakomerno uravnoteženo. Ko iščemo vzorec, ki bo imel najboljše ujemanje, se preprosto sprehodimo od korena do lista in na vsakem koraku izračunamo Hammingovo razdaljo. Na koncu izberemo polje z najnižjo vrednostjo, katero predstavlja najboljše ujemanje vzorcev.

4.4.2 Wildova metoda primerjanja

Wildov sistem uporablja bolj zapleteno metodo za primerjavo vzorcev. Metoda deluje na principu normalizirane korelacije nad zajetim vzorcem in vzorcem v podatkovni bazi. Normalizirana korelacija zajame isti vzorec informacije na točki enako kot standardna korelacija s tem, da normalizirana korelacija preprečuje razlikovanje lokalnega vzorca zaradi svetlobnih razlikovanj, kateri so problematični pri navadni korelaciji. Korelacija se izvaja na manjših blokih (8x8) pikslov v štirih frekvenčnih prostorih piramide. Poznamo 2 vrsti piramid, "lowpass" in "bandpass". Z uporabo "lowpass"piramide, sliko spremenimo tako, da originalno sliko pod-vzorčimo. Nad sliko izvedemo funkcijo glajenja in nato prepolovimo resolucijo slike. Primer takšne piramide je Gaussianova piramida. Primer "bandpass" piramide pa je Laplacova piramida, kater ase razlikuje od "lowpass" piramid le v tem, da namesto, da hranimo celotne podvzorčene slike, hranimo le razlike med njimi. Tako lahko iz podvzorci rekonstruiramo originalno sliko. Ta tehnika podvzorčenja je

uporabljena tudi za kompresijo slik.

Wildova metoda primerjanja vzorcev uporablja frekvenčni prostor Laplacove piramide. Korelacija se izvede za vsak blok v frekvenčnem prostoru. To privede do več rezultatov istega bloka. Te rezultate nato združimo z izračunom mediane in tako dobimo končno vrednost bloka. Uporaba frekvenčnih prostorov omogoča preprečevanje morebitnih napak, ki so nastale pri zajemanju vzorcev zaradi popačenja slike šarenice.

Če sta $p1(i,j)$ in $p2(i,j)$ tabeli vzorcev slike velikosti $n * m$. Parametra μ in θ predstavljata povprečni in standardni odklon intenzitete. $\mu = (1/n*m)*\Sigma_n*\Sigma_m(p(i,j))$ $\theta = \sqrt{(1/n*m)*\Sigma_n*\Sigma_m(p(i,j)-\mu)^2}$, potem je normalizirana korelacija slike p1 in p2 $NC = \frac{\Sigma_n*\Sigma_m(p_1(i,j)-\mu_1)(p_2(i,j)-\mu_2)}{n*m*\theta_1*\theta_2}$. Robustnost te metode je prav v tem, da se povprečni intenzitenti odklon odšteje v števcu medtem, ko se standardni odklon pojavi v imenovalcu funkcije. S tem omogočimo, da se sliki čimbolj približata v svetlostnih razlikah.

4.4.3 Ostale metode primerjanja

Poleg že omenjenih metod, se lahko za primerjanje vzorcev uporabljajo tudi druge matematične funkcije. Poznamo funkcije za računanje razdalje med točkami na grafih. Te funkcije se lahko uporabi tudi v našem primeru, ko primerjamo razliko med dvema nizoma bitov. Tako lahko uporabimo N-dimenzionalno funkcijo za izračun Manhattenske, Euclidske ali pa Minkowske razdalje. Vse delujejo na podoben način in sicer se razdalja izračuna po naslednji formuli: $d(p,q) = \Sigma_n(|p_i - q_i|^c)^{1/c}$, kjer sta p in q vzorca katera primerjamo, c predstavlja dimenzijo vzorca. Če imamo c, ki je enak 1 ali 2, postane Minkowska razdalja enaka ostalima dvema.

4.5 Odločanje o rezultatu

Kot zadnje opravilo pri prepoznavanju šarenice je potrebno rezultat, ki smo ga dobili pri primerjanju vzorcev še določiti v dve skupini. Tisti, kateri so veljavni in tisti kateri so lažni. V ta namen moramo določiti mejo med

veljavnimi rezultati in lažnimi.

4.5.1 Daugmanova metoda odločanja

Sistem uporablja mejo za normalizirane Hamingove razdalje primerjanih vzorcev. Razdalje manjše od izbrane meje bodo vzete, kot veljavni rezultat. Za določanje meje odločanja je uporabljen binomski izrek iz kombinatorike. $p(k) = \binom{n}{k} * p^k(1-P)^{n-k}$ Dejansko vrednost dobimo tako, da na določenem vzorcu izvedemo metodo primerjanja (Hammingovo razdaljo). Vzorec mora vsebovati na primer, 100 pozitivnih rezultatov in 100 negativnih. Nato izračunamo povprečno vrednost in tako dobimo mejo med dobrimi in slabimi rezultati.

4.5.2 Wildova metoda odločanja

Wildov sistem mora pri odločanju združiti vse štiri rezultate frekvenčnih prostorov Laplac-ove piramide v enotni rezultat. Te rezultate moramo združiti v tako kombinacijo mer-jenih spremenljivk, da bo maksimalno ločila vnaprej določene skupine in da bo napaka pri uvrščanju najmanjša. To omogoča Diskriminantna analiza. Pri diskriminantni analizi gre za iskanje tistih razsežnosti, ki kar najbolj pojasnjujejo razlike med skupinami.

Če imamo n d-frekvenčnih vzorcev Q in od tega n_a pravilnih in n_i slabih vzorcev z diskriminanto dobimo ϕ, ki je razmerje, ki tvori največje razlike med dobrimi in slabimi rezultati. Za vsako skupino (dobre in slabe) izračunamo varianco $S_a = \Sigma_q((q-\mu_a)(q-\mu_a)^t)$, kjer je t število frekvenčnih prostorov in μ je povprečna vrednost med izbranimi frekvenčnimi prostori. Nato lahko izračunamo razmerje $\phi = S_\phi^{-1} * (\mu_a - \mu_i)$, kjer je $S_\phi = S_a + S_i$ skupna vrednost varianc za dobre in slabe skupine.

Poglavje 5

Rezultati omenjenih metod

Oba sistema sta se v praksi dobro izkazala. Daugmanov sistem je bil implementiran tako, da je imel možnost opravljanja treh funkcij. Funkcijo "zajema", katera je zajela sliko in iz nje sprocesirala zapis, ki se nato hrani v bazi podatkov za nadaljno rabo. Funkcijo verifikacije, ki primerja zajeto sliko s točno določenim zapisom v bazi (ena na ena primerjanje) ter funkcijo "identifikacje", katera primerja zajeto sliko s celotno bazo podatkov. Obe funkciji "zajema" in "verifikacije" svoje delo opravita v manj kot sekundi, medtem, ko funkcija identificiranja v istem času lahko opravi preverjanje nad 4000 podatki. Daugmanov algoritem se je dobro uveljavil in je bil razvit tudi za komercialno uporabo.

Wildov sistem je bil implementiran tako, da omogoča le dve funkciji žajema"in "verificatije". Obe funkciji sta počasnejši od Daugmanovih, svoje delo opravita v slabih 10 sekundah. Je pa možno sistem spremeniti in s tem izboljšati delovanje. Wildov sistem se ni tako dobro uveljavil v uporabo in tudi ni na voljo v komercialnem sistemu.

5.1 Daugmanov sistem

Daugmanov algoritem uporablja že omenjeno Hammingovo razdaljo za primerjanje vzorcev, za katero izračunamo rang na podlagi vzorca. Manjši ko je

rang, manjša je možnost napake. Pri tem je potrebno vedeti, da je pri manjši vrednosti ranga, večja možnost, da pride do "lažno pozitivnega" rezultata. To pomeni, da bo pravilni rezultat napačen in zaradi tega prikazan kot napaka. Spodaj je tabela 8.3 rezultatov testiranja [1] Daugmanovega sistema za različne vrednosti rangov.

HD rang	možnost napake
0.26	1 na 10^{13}
0.27	1 na 10^{12}
0.28	1 na 10^{11}
0.29	1 na $13 * 10^{9}$
0.30	1 na $1.5 * 10^{9}$
0.31	1 na 185mil
0.32	1 na 26mil
0.33	1 na 4mil
0.34	1 na 690 000
0.35	1 na 133 000

Tabela 5.1: Daugmanov sistem - učinkovitost iskanja

Pri testiranju učinkovitosti iskanja iz baze [5] je imel Daugmanov sistem 0.08 odstotno stopnjo napake, kar pomeni, da je prepoznal 99.9 odstotkov šarenic. Sistem so testirali nad podatkovno bazo, katera je vsebovala 4258 različnih slik šarenic.

Časovna zahtevnost algoritma je dokaj majhna, saj celotno delo opravi v manj kot sekundi časa. Sistem so testirali z 300MHz procesorjem, kateri je zmožen opraviti 100 000 primerjanj različnih šarenic na sekundo in preko milijon primerjanj v 1.7 sekunde na 2GHz procesorju. V spodnji tabeli 5.2 je prikazana časovna odvisnost za posamezno operacijo sistema.

operacija	čas
lokalizacija	90msec
segmentacija	12msec
zaznavanje vek	93msec
zaznavanje trepalnic	78msec
kodiranje vzorca	102msec
primerjava vzorca	10μsec

Tabela 5.2: Daugmanov sistem - časovna zahtevnost

5.2 Wildov sistem

Pri testiranju [13] so uporabili 50 šarenic naključnih posameznikov. Od tega jih 10 predstavlja napačne rezultate, ostalih 40 pa pravilne. Za vsakega posameznika so vzeli po 5 slik, tako je skupaj 400 slik, ki so veljavne. Vseh 400 slik so primerjali s tistimi (50) v bazi in sistem je za vse slike našel pravilni rezultat. Testirali so tudi tako, da so primerjali levo in desno šarenico. Tudi tukaj je test pokazal 100 odstotni rezultat.

Pri testiranju učinkovitosti iskanja iz baze [5], je Wildov sistem pokazal 1.7 odstotno stopnjo napake pri 60 slikah, medtem ko Daugmanov sistem le 0.08 odstotka pri 4258 slikah. Oba sistema sta dovolj dobra za uporabo prepoznavanja identitete oseb.

Poglavje 6

Predstavitev orodij za izdelavo sistema

Za izdelavo svojega sistema za prepoznavanje očesne šarenice smo uporabili programski jezik in okolje MATLAB. Delovanje sistema je testirano z uporabo slik iz zbirke CASIA v4, katera je namenjena razvoju biometričnih sistemov.

6.1 MATLAB

Matlab [8] je programsko okolje in programski jezik četrte generacije. Matlab omogoča računanje z matrikami, programirnaje in izvajanje algoritmov, izdelovanje uporabniških vmesnikov ter podatkovno vizualizacijo. Omogoča povezovanje z drugimi programskimi jeziki, kot so: C, C++, C#, Java, Python in Fortran. Matlab v praksi uporabljajo različni strokovnjaki iz področij procesirnaj slik in signalov, komunikacije, krmilnih sistemov za industrijo, pametna zasnova omrežja, robotika in v računalniških financah. Cleve Moler profesor na mehiški univerzi je ustvaril MATLAB leta 1970, da bi pomagal svojim študentom. Leta 1984 so MATLAB prepisali v jezik C in ga komercialno uveljavili v uporabo. Leta 2004 je obsegal kar štiri milijone uporabnikov z različnih področij industrije.

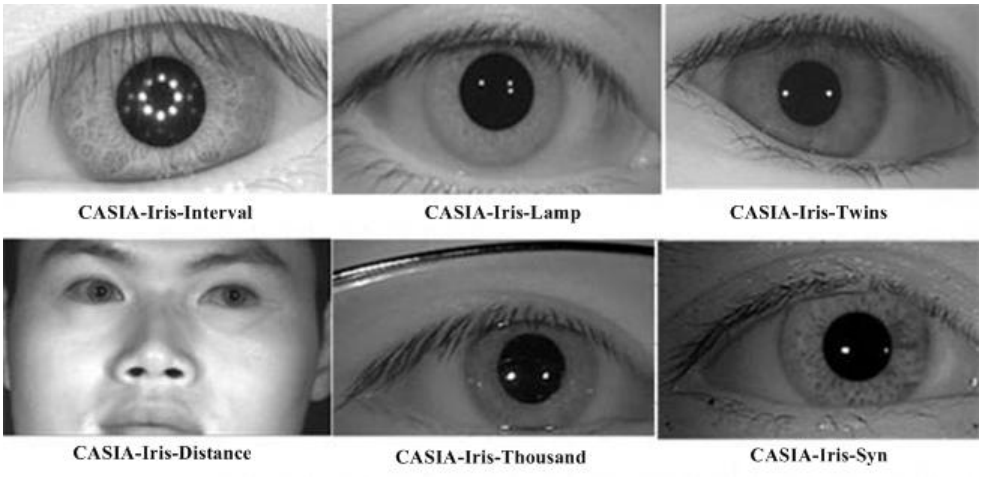
6.1.1 Zakaj Matlab

Glavna lastnost zaradi katere je MATLAB uporabljen, je hitro programiranje. Matlab vsebuje ogromno funkcij, katere so na voljo in jih ni potrebno ustvarjati vedno znova. Najdi unikatni element v zbirki: ***unique(list);*** konvolucija n-dimenzionalne tabele: ***convn(A);*** izračun trajanja izvajanja kode: ***tic; koda; toc;*** grafični vmesnih za izrezovanje slik: ***imcrop(im);*** in še veliko ostalih funkcij. Programski jezik, ki se najbolje približa Matlabu je Python in Octave. Sintaksa programskega jezika Octave je zelo blizu sintaksi Matlaba s tem, da je Octave brezplačen. Problem pri Pythonu je prav v tem, da je funkcije težko najti, nato je potrebno uvoziti pakete, ki vsebujejo te funkcije in še paketi niso vedno kompatibilni drug z drugim. Prav tako vsebuje funkcije za izrisovanje grafov in vizualizacijo podatkov. Ker so funkcije že vnaprej ustvarjene, uporabnik Matlaba ne potrebuje razmišljati, kako bo ustvaril določen del kode ampak le uporabi funkcijo in s tem tudi omogoči lažjo berljivost kode. Vse funkcije so dokumentirane in jih tako hitro in enostavno poiščemo ter poizvemo o njihovi uporabi. Slaba stran Matlaba pa je v tem, da ni prosto dostopen.

6.2 CASIA - zbirka slik

CASIA [7] je zbirka slik šarenic, ki je namenjana razvoju sistemov za prepoznavanje šarenic. Deluje od leta 2002 in je trenutno v 4 verziji, ki zajema 54.000 slik od več kot 1800 osebkov. Zbirka je sestavljena iz 6 sklopov slik (slika 6.1), namenjene za vse možne situacije pri prepoznavanju šarenice. Prvi sklop vsebuje detajlne slike šarenice, namenjene predvsem branju teksture iz slik. Drugi sklop zajema slike, ki so bile zajete z ročno kamero in so namenjene za analiziranje problemov pri segmentaciji in normalizaciji zaradi nelinearnosti in rotacijskih napak na sliki. V tretjem sklopu so zajete slike dvojčkov z namenom, da dokažemo unikatnost šarenice tudi pri dvojčkih. V četrti zbirki so zajete slike na določeni razdalji, za prikaz, če sistem deluje tudi na slikah, ki so bile zajete z različne oddaljenosti od kamere. Zadnja dva

sklopa pa vsebujeta velik nabor različnih slik šarenic različnih pigmentov od oseb različnih starosti. Namen teh sklopov je testiranje učinkovitosti sistemov in unikatnosti šarenic. Opis posameznega sklopa in število vsebovanih slik je predstavljen v tabeli 6.1.



Slika 6.1: CASIA zbirka slik

Sklop	Iris-Interval	Iris-Lamp	Iris-Twins	Iris-Distance	Iris-Thousand	Iris-Syn
Št. oseb	249	411	200	142	1000	1000
Št. slik	2639	16212	3183	2567	20000	10000
lastnosti	slike z razločno teksturo šarenice	rotacijske napake, svetlobni odsevi	slike dvojčkov	slike z razdalje	zbirka z več kot 1000 osebkov	združena zbirka

Tabela 6.1: Zbirka slik CASIA

Poglavje 7

Predstavitev izdelane metode

Izdelali smo sistem za prepoznavanje očesne šarenice v programskem jeziku Matlab. Sistem je sestavljen iz sedmih funkcij. Sistem vzame sliko šarenice kot vhodni parameter in vrne kodni zapis in masko v obliki 50x250 tabele. Maska služi kot indikator, katera polja v zapisu so pokvarjena in katera so dobra za uporabo pri primerjavi vzorcev. Pri izdelavi smo uporabili nekatere že obstoječe funkcije, ki jih vsebuje MATLAB. Pri izdelavi sistema smo uporabljali slike iz zbirke CASIA v4 - Iris Interval.

7.1 Funkcija segmentacije

Funkcija za segmentacijo slike šarenice temelji na Wildovi metodi. Za lociranje robov med šarenico, beločnico in zenico smo uporabili Houghovo transformacijo za iskanje krogov na slikah ter Canny edge metodo zaznavanja robov na sliki. Uporabili smo obstoječi matlabovi funkciji "edge"[14] in "imfindcircles"[15]. Houghova transformacija poišče vse možne kroge na sliki, s tem dobimo veliko število krogov. Kot rezultat dobimo tudi matriko z vrednostmi za vsak krog, katera predstavlja število zadetkov za ta krog. Večja kot je številka, bolj verjetno je, da se tam nahaja krog. Funkcija vrne tabelo s 4 vrednostmi; središče zenice ter radij in središče šarenice in njen radij.

Funkcija prejme izvorno sliko šarenice, na podlagi katere naredi sliko z

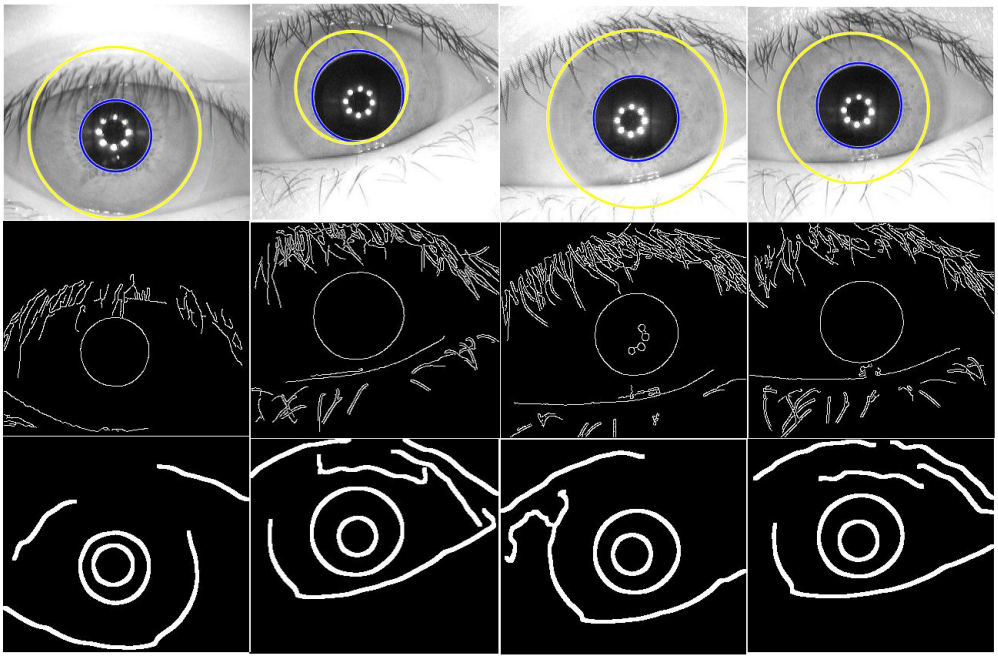
robovi za iskanje zenice. Z zviševanje meje intenzitete (tresholding) nad izvorno sliko, kar hitro dobimo lep rob med šarenico in zenico. Na Sliki 7.1 v drugi vrstici je prikazana slika robov za iskanje zenice. Pri iskanju zenice ni bilo potrebno nobenih izboljšav, saj je postopek našel zenico na vseh testnih slikah.

Za iskanje roba med šarenico in beločnico je postopek zelo podoben. Prav tako izdelamo sliko robov s tem, da moramo tukaj uporabiti glajenje slike, saj s tem povečamo razliko med belim delom in šarenico. Na sliki 7.1 v tretjem stolpcu lahko vidimo robno sliko za iskanje roba med šarenico in beločnico. Houghova transformacija tukaj ni dala najboljšega rezultata, saj je bilo veliko najdenih krogov bolj verjetnih, kot pa tisti, ki je bil dejansko pravilni rob. Zaradi pomanjkljivosti pri iskanju, je bilo potrebno med vsemi dobljenimi krogi poiskati najboljšo rešitev. Za iskanje najboljše rešitve, smo uporabili določene parametre: velikost kroga, oddaljenost središča od središča zenice ter število zadetkov za iskani krog. Za vse kroge, ki so bili nad določeno mejo zadetkov, smo nato poiskali tisti krog, ki ima največji polmer in katerega središče je kar najbližje središču zenice.

Kot lahko vidimo na sliki 7.1 v prvi koloni, kljub iskanju najboljše možne rešitve za zunanji rob, še vedno pri nekaterih slikah funkcija ni našla pravilnega roba.

7.2 Funkcija za lociranje vek in trepalnic

Ko imamo locirano območje zenice iz koraka segmentacije, je potrebno na tem območju odstraniti veke in trepalnice, če le te prekrivajo šarenico. Veki in trepalnice se običajno nahajajo v zgornjem in spodnjem delu območja šarenice. Veke so horizontalne krivulje, trepalnice pa pretežno vertikalne. Funkcija lociranja pretežno temelji na Wildovi metodi, območje kjer se nahajajo trepalnice in veka pa označimo po Daugmanovi metodi. Za iskanje vek in treplanic smo uporabili Matlabovo linearno Houghovo transformacijo imenovano “houghline” [16] in enako kot pri segmentaciji Canny edge metodo



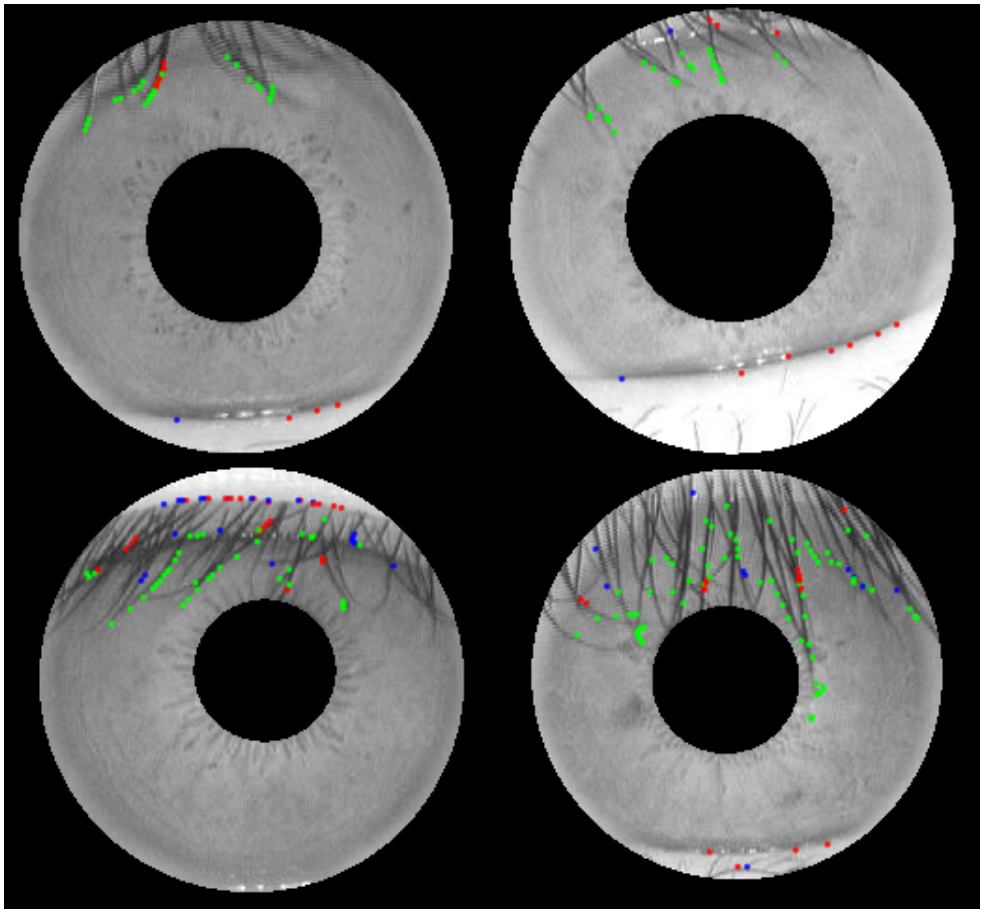
Slika 7.1: Segmentacija slike šarenice

za iskanje robov na sliki.

Linearna Houghova transformacija na sliki poišče vse možne točke, ki se nahajajo na črti. Teh točk je mnogo, zato jih je bilo potrebno omejiti. Postavili smo mejo 200 točk, katere tvorijo najbolj očitne črte. Ker so te črte v 2D prostoru, imajo lahko različni smerni kot. Ker vemo, da so veke običajno horizontalne in trepalnice vertikalne, lahko ta kot omejimo na določene vrednosti in s tem dobimo le točke, ki potencialno tvorijo veke in trepalnice. Mejo, ki smo jo vzeli za iskanje trepalnic je bila med 40 in -40 stopinj v primeru, da je izhodišče v sredini zgornjega roba slike. Kot za iskanje vek smo omejili na 2 intervala in sicer koti med 65 in 115 stopinj ter interval med -65 in -115. Na sliki 7.2 so prikazane dobljene točke. Zelene točke predstavljajo trepalnice, modre in rdeče pa spodnjo in zgornjo veki na sliki.

Tako dobimo vse črte, ki potekajo pod določenimi smernimi koti in potencialno predstavljajo veke oziroma trepalnice. Ko imamo zbrane vse tiste točke, katere potencialno tvorijo veko oziroma trepalnico, je potrebno izmed vseh točk izbrati le najbolj pomembni točki. Ti dve točki tvorita zgornjo in

spodnjo mejo, kjer veke ali trepalnice prekrivajo šarenico. Zgornjo in spodnjo mejo dobimo tako, da iz zbirke točk vzamemo tisto točko, ki se nahaja najnižje v zgornjem delu šarenice nad središčem zenice in točko, ki se nahaja najvišje v spodnjem delu šarenice pod središčem zenice.



Slika 7.2: Iskanje vek in trepalnic

Funkcija vrne le vrednosti x in y za točki, ki mejita na območje šarenice, kateri vsebujeta trepalnice ali veke.

7.3 Funkcija za odstranjevanje nepomembnih delov slike

Podatke, ki smo jih pridobili pri segmentaciji in lociranju vek in trepalnic, sedaj uporabimo v funkciji za odstranjevanje nepomembnih delov slike šarenice. Funkcija iz osnovne slike šarenice izreže le tisti del, kje se nahaja območje šarenice, ki ni "pokvarjeno". Za odstranitev področij zunaj območja šarenice, potrebujemo podatke pridobljene pri segmentaciji in sicer središče in premer zenice ter šarenice.

Za vse točke, ki se nahajajo znotraj kroga zenice in zunaj kroga šarenice nastavimo barvne vrednosti na NULL. Da ugotovimo, kdaj se neka točka nahaja znotraj kroga lahko preprosto uporabimo Pitagorov izrek $k1^2 + k2^2 = h^2$. Vsaka točka, ki se nahaja v območju zenice bo ustrezala pogoju $(|X - Xpc|)^2 + (|Y - Ypc|)^2 <= rp^2$ in točke zunaj območja šarenice pa pogoju $(|X - Xic|)^2 + (|Y - Yic|)^2 >= ri^2$.

Iskanje točk znotraj krogov lahko malenkost pohitrimo z manj izračunavanja, tako, da krogu očrtamo in včrtamo kvadrat. S tem za večino točk ne potrebujemo operacije potenciranja in enostavno preverimo pogoj $|X - Xpc| <= rp$ oziroma $|Y - Ypc| <= rp$ za točke, ki se nahajajo znotraj včrtanega kroga zenice in pogoj $|X - Xic| >= ri$ oziroma $|Y - Yic| >= ri$ za točke, ki se nahajajo zunaj očrtanega kvadrata zenice. Z uporabo teh pogojev zadostimo že večini točk, vendar pa so med krogom in kvadratom še nekateri deli, ki jih nismo pokrili. Te dele pokrijemo z že omenjenim pitagorovim izrekom, pri tem postopku smo za večino točk zmanjšali čas izračunavanja, ki se porabi za izračun potenciranja.

Na dobljenem območju šarenice moramo odstraniti še zgornjo in spodnjo mejo, kjer je šarenica "pokvarjena". V ta namen uporabimo točki, ki smo jih dobili pri izvedbi funkcije za lociranje vek in trepalnic. Vse dobljene točke znotraj območja šarenice tako omejimo še na zgornjo in spodnjo mejo. Vsem točkam, ki se nahajajo nad zgornjo mejo in vsem, ki se nahajajo pod spodnjo mejo nastavimo barvno vrednost na NULL.

Tako kot končni rezultat dobimo le točke, ki se nahajajo med omenjenimi mejami.

7.4 Funkcija normalizacije šarenice

Šarenico ločeno od ostalega dela je potrebno pretvoriti iz polarne oblike v kartezično (slika 7.3). Funkcija za normalizacijo šarenice temelji na Daugmanovi metodi imenovani Daugman rubber sheet. Rezultat, ki smo ga dobili pri odstranjevanju neuporabnih delov uporabimo za normalizacijo. Poleg slike šarenice, potrebujemo še radij šarenice in radij zenice.

Slika 7.3: Normalizirana šarenica

Vsako polarno točko v območju šarenice preslikamo v kartezično točko tako, da preslikamo vrednost točke $K(x, y)$, kjer je kordinata x na polarni sliki $x = r * \cos(\theta)$ kordinata $y = r * \sin(\theta)$. Pri preslikovanju točk uporabimo bilinearno interpolacijo za tiste točke, katerih kordinate niso cela števila. Za vsako točko, katera nima kordinat, ki so cela števila, izračunamo vrednost nove točke z uporabo interpolacije. Za vse točke, katerih kordinata Y je realno število naredimo interpolacijo v smeri X. Vrednost nove točke dobimo $V(x, y) = V(X, Yf) + (Y - Yf) * (V(X, Yc) - V(X, Yf))$, kjer Y predsta-

vlja kordinato, ki je realno število. Parameter Yf je spodnja meja realnega števila, Yc je zgornja meja realnega števila in X predstavlja osnovno kordinato, ki je celo število. V primeru, da je koordinata X realno število, dobimo vrednost nove točke tako, da interpoliramo v smeri Y $V(x,y) = V(Xf,Y)+(X-Xf)*(V(Xc,Y)-V(Xf,Y))$. V primeru, ko sta obe koordinati realno število, izvedemo interpolacijo v obeh smereh x in y tako, da pomnožimo vektorja $V = A*w$, kjer A predstavlja vektor, katerega izračunamo po naslednji formuli $A = \begin{bmatrix} Xf & Yf & Xf*Yf & 1 \\ Xf & Yc & Xf*Yc & 1 \\ Xc & Yf & Xc*Yf & 1 \\ Xc & Yc & Xc*Yc & 1 \end{bmatrix} * \begin{bmatrix} V(xf,yf) \\ V(xf,yc) \\ V(xc,yf) \\ V(xc,yc) \end{bmatrix}$ in w vektor $w = \begin{bmatrix} X & Y & X*Y & 1 \end{bmatrix}$

Rezultat funkcije je normalizirana šarenica v kartezični obliki (dvodimenzionalna tabela) velikosti $r \times \theta$.

7.5 Funkcija za kodiranje vzorca

Iz normalizirane šarenice je potrebno sestaviti kodni zapis. Funkcija za kodiranje vzorca iz teksturnih značilnosti šarenice sestavi unikatni kodni zapis. Funkcija deluje tako, da za vsako točke normalizirane šarenice izračuna vrednost. Tako dobimo tabelo vrednosti, katera tvori naš kodni vzorec. Poleg kodnega vzorca potrebujemo tudi masko vzorca, katera nam pove, kateri deli vzorca so dobri za uporabo in kateri ne. Za kodiranje vzorca smo uporabili metodo Gabor filter.

Za normalizirano šarenico smo izračunali rezultat Gabor filtra pod različnimi koti. Za vsako sliko smo izračunali vrednost pod 8 smernimi koti (interval od 0 do 160 in stopnjo povečevanja 22.5). Kot rezultat dobimo 8 tabel z vrednostmi. Za vsako soležno vrednost v tabeli izračunamo povprečno vrednost vseh 8 rezultatov Gabor filtra. Kot rezultat dobimo dvodimenzionalno tabelo vrednosti, katere je potrebno kvantizirati, tako da dobimo manjši nabor vrednosti, katere bomo nato primerjali.

Poleg rezultata Gabor filtra je potrebno v tem koraku določiti masko vzorca. Maska vzorca določa, kateri del vzorca je dober za uporabo. Območje, katerega smo izločili v koraku odstranjevanja nepomembnih delov, kjer so se nahajale trepalnice ali veki, označimo kot pokvarjen del vzorca. Maska predstavlja dvodimenzionalno tabelo enake velikosti, kot vzorec z vrednostmi 0 in 1.

Končni rezultat predstavlja dve tabeli z vrednostmi, prva je dejanski digitalizirani vzorec šarenice, druga pa maska, ki določa, kateri deli vzorca so uporabni za primerjanje.

7.6 Funkcija za primerjanje vzorcev

Primerjanje vzorcev temelji na izračunu Hamingove razdalje. Za izračun razlike med vzorcema, potrebujemo štiri podatke in sicer obstoječ vzorec in masko iz podatkovne baze, ter trenutno pridobljeno masko in vzorec. Maska vzorca je dvodimenzionalna tabela, ki vsebuje vrednosti 0 in 1. Za vse vrednosti v tabeli, ki so nastavljene na 0 pomeni, da je na tem delu pokvarjen del šarenice in ga ne smemo vključiti v izračun. Za vse vrednosti v maski, ki so enake 1 primerjamo istoležne vrednosti v obstoječem in novem vzorcu. Za vsaki vrednosti, ki se razlikujeta, povečamo števec razlik. Na koncu število razlik normaliziramo, tako, da jo delimo s številom primerjanih vrednosti v vzorcu.

Dobimo procentualno odstopanje vzorcev, na podlagi katerega se nato odločimo ali gre za šarenico iste osebe. Mejo, ki loči pravilne rezultate od nepravilnih, dobimo tako, da naredimo statistični vzorec in na podlagi tega poiščemo primerno mejo.

Poglavje 8

Rezultati izdelane metode

Svojo rešitev sistema smo testirali z uporabo slik iz zbirke CASIA v4. Izbrali smo naključnih 100 slik kot testni vzorec. Na podlagi tega vzorca smo za vsako funkcijo izračunali čas izvajanja in če je možno tudi stopnjo uspeha.

8.1 Segmentacija

Sistem opravi postopek segmentacije v povprečno 0,5 sekunde. Za lokalizacijo robov šarenice sistem potrebuje približno 0.25 sekunde, za iskanje vek in trepalnic 0.05 sekunde. Preostanek časa sistem porabi pri odstranjevanju nepomembnih delov na sliki.

Stopnjo uspeha smo pri segmentaciji razdelili v 2 dela. Prvi del zajema uspešnost lociranja robov šarenice, drugi pa lociranje vek in trepalnic. Za vzorec 100 slik je segmentacija locirala rob šarenice z uspešnostjo 86 odstotkov. Za zaznavanje vek in trepalnic je bila uspešnost 82 odstotkov na izbranem vzorcu.

Pri iskanju robov šarenice je sistem našel rob med šarenico in zenico pri vsaki sliki, kar pomeni uspešnost 100 odstotkov. Problem je bil zunanji rob med šarenico in beločnico, za katerega smo naredili funkcijo iskanja najboljšega ujemanja. Funkcija deluje v redu, kot je razvidno iz rezultata, vendar bi se dalo rezultat izboljšati. Tudi glede časovne zahtevnosti bi se

dalo rezultat nekoliko izboljšati.

8.2 Normalizacija in kodiranje vzorca

Pri normalizaciji in kodiranju vzorcev smo testirali le časovno zahtevnost, saj se za funkciji ne da izračunati stopnjo uspeha. Čas izvajanja normalizacije je v povprečju 0.085 sekunde. Pri tej funkciji bi se čas izračunavanja lahko nekoliko izboljšal z uporabo pogojev omenjene v tretjem odseku.

Sistem za izračun vzorca porabi med 0.2 in 0.35 sekunde. Čas izvajanja je odvisen od maske vzorca, saj tiste dele, kateri so pokvarjeni ne upoštevamo pri izračunavanju vzorca.

8.3 Primerjava vzorca

Za izračun Hammingove razdalje s katero primerjamo razliko med vzorci, sistem porabi med 0.0008 in 0.001 sekunde. Tudi pri računanju razlike je hitrost izvajanja odvisna od maske vzorcev, saj pokvarjene dele preprosto preskočimo in s tem zmanjšamo število izračunavanj.

8.4 Rezultati učinkovitosti sistema

Učinkovitost je bila testirana na dva načina. Prvi test pokaže unikatnost šarenice, pri primerjanju šarenic iste osebe in šarenic različnih oseb. Drugi test zajema, kako učinkovit je sistem pri iskanju določene osebe med drugimi osebami.

Za prvi test smo uporabili slike šarenic od 240 oseb. Končni rezultat zajema 1040 slik šarenic in 520 primerjav. Za kodiranje vzorca smo uporabili navadne Gabor filtre, kot je predstavljeno v odseku 7.5 in za primerjavo še Log-Gabor filtre.

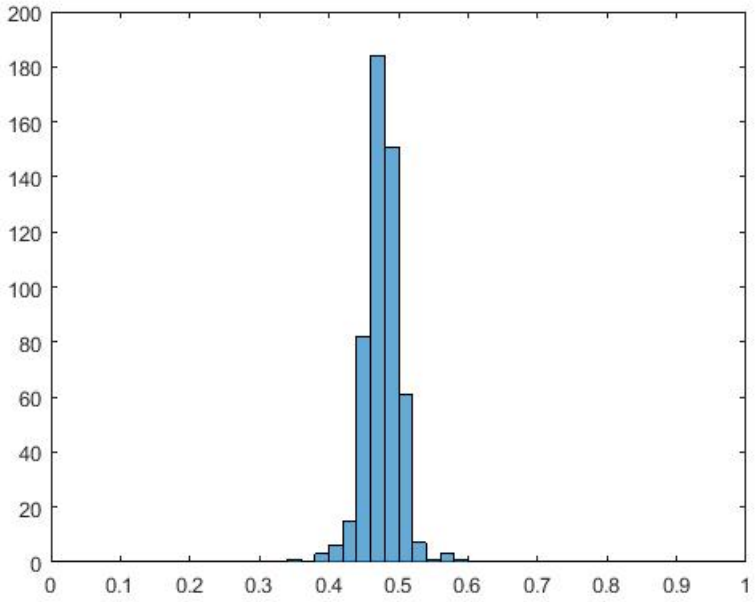
8.4.1 Rezultati unikatnosti šarenice

Rezultat unikatnosti šarenice z uporabo navadnih Gabor filtrov, je pokazal slabe rezultate pri razlikovanju med primerjanjem šarenic istih (Intra-class) in različnih oseb (Inter-class).

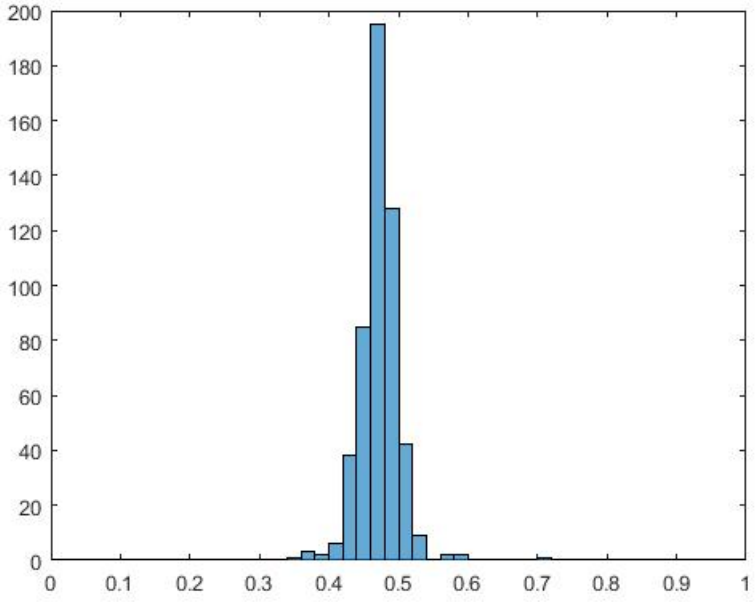
Kot lahko vidimo (sliki 8.1 in 8.2) se razreda skoraj popolnoma prekrivata. Sistem ima visoko možnost napačne odločitve, zato ni zanesljiv za uporabo. Prav tako, nimamo neke očitne meje med razredoma, da bi lahko določili mejo med veljavnimi in neveljavnimi rezultati.

Sistem smo testirali še z uporabo Log-Gabor filtrov. Implementacija Log-Gabor filtra, ki smo jo uporabili pri testiranju, je odprtokodna rešitev. Implementiral jo je Libor Masek [17]. Pri testiranju unikatnosti z uporabo Log-Gabor filtrov, je bil rezultat veliko boljši. Razlika med razredoma je očitna. Večina primerjanj istih šarenic, vrne rezultat med 0.1 in 0.4, primerjanje različnih šarenic pa med 0.4 in 0.6 (slika 8.3 in 8.4). Rezultat še vedno ni popoln, saj se določeni deli še vedno prekrivajo. Na nepopolnost ima nekaj vpliva tudi nepopolno delovanje segmentacije šarenice in lociranje vek in trepalnic, kot tudi normalizacija šarenice, ki trenutno deluje, kot da sta zunanji in notranji rob vedno linearno poravnana.

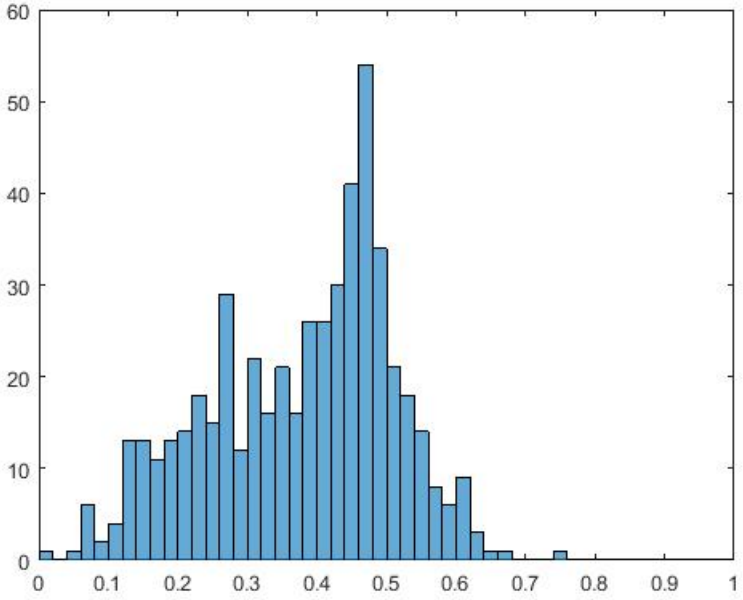
Tabeli 8.1 in 8.2 prikazujeta možnost pravilnih FAR in nepravilnih FRR napak. Tudi tukaj se vidi slabše delovanje pri testiranju z navadnimi Gabor filtri.



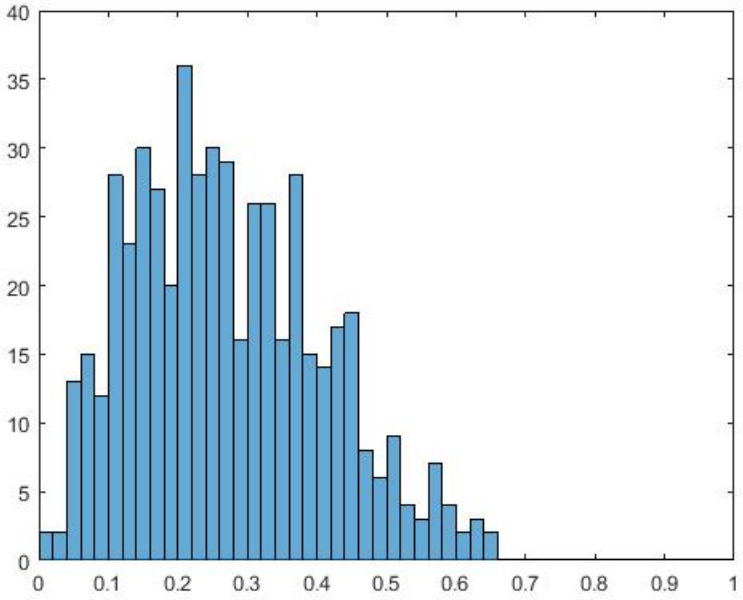
Slika 8.1: Inter-class z uporabo Gabor filtrov



Slika 8.2: Intra-class z uporabo Gabor filtrov



Slika 8.3: Inter-class z uporabo Log-Gabor filtrov



Slika 8.4: Intra-class z uporabo Log-Gabor filtrov

Stopnja meje	FAR(%)	FRR(%)
0.20	0	100
0.25	0	100
0.30	0	100
0.35	0	100
0.40	57	98
0.45	55	82
0.50	96	16

Tabela 8.1: Rezultat stopnje napak z uporabo Gabor filtra

Stopnja meje	FAR(%)	FRR(%)
0.20	13	69
0.25	17	52
0.30	26	41
0.35	35	30
0.40	48	20
0.45	62	12
0.50	87	6

Tabela 8.2: Rezultat stopnje napak z uporabo Log-Gabor filtra

8.4.2 Rezultati učinkovitosti iskanja

Pri testiranju učinkovitosti iskanja šarenice med drugimi šarenicami različnih oseb, smo uporabili 100 slik od 100 naključnih oseb. Za 25 oseb smo nato vzeli še dodatno sliko (25 slik). Vsako posamezno sliko izmed 25 izbranih slik smo primerjali s 100 slikami. Stopnjo meje za iskanje pravilnih rezultatov smo vzeli 0.4. Tabela 8.3 prikazuje rezultat testiranja. Kot lahko vidimo, je sistem za 7 šarenic našel napačno ujemanje (FAR), kar je slabih 30% in za 2 šarenici je bil rezultat pravilen, vendar je meja presegala stopnjo 0.4, kar

se šteje kot napačna zavrnitev (FRR). Za ostalih 16 (64%) šarenic pa je bil rezultat pravilen in pod iskano mejo.

Stopnja	**Skupaj**	**POS**	**FAR**	**FRR**	**FAR(%)**	**FRR(%)**
0.4	25	16	7	2	28	8

Tabela 8.3: Rezultat učinkovitosti iskanja

Poglavje 9

Nadaljnje raziskave

Na predstavljenem sistemu, bi bilo možno veliko popraviti. Metoda segmentacije ni delovala povsem odlično, saj najdeno območje ni vedno povsem prekrivalo šarenico ali pa je zajemalo območje izven šarenice. Prav tako, je iskanje vek in trepalnic možno nadomestiti z bolj elegantno rešitvijo, ki ne odreže celotnega dela, kjer se nahajajo trepalnice ampak odstrani vsako trepalnico posebej.

Pri postopku normalizacije, bi bilo potrebno nadgraditi delovanje, da bi se izognili nelinearnosti notranjega in zunanjega roba šarenice. Pri trenutni metodi predpostavljamo, da sta kroga vedno centrično poravnana.

Pri kodiranju vzorca, bi bila potrebna implementacija Log Gabor Filtra, saj trenutna metoda, ki uporablja navadni Gabor filter ne deluje najbolje in s tem močno vpliva na delovanje celotnega sistema pri primerjavi vzorcev.

Pri metodi primerjanja vzorcev, bi bila dobrodošla le hitrejša implementacija izračunavanja Hammingove razdalje.

Poglavje 10

Zaključek

10.1 Povzetek dela

V delu je bilo podrobno predstavljeno delovanje obstoječih sistemov za prepoznavanje očesne šarenice in na podlagi teh sistemov izdelana lastna rešitev sistema.

Predstavljena je bila izdelana metoda za avtomatsko segmentacijo očesne šarenice, katera iz slike odstrani vse nepotrebne dele in loči območje, kjer je šarenica pokvarjena zaradi prekrivanja vek in trepalnic. Metoda avtomatske segmentacije uporablja krožno Houghovo transformacijo za zaznavanje zenice in šarenice ter linearno Houghovo transformacijo za zaznavanje vek in trepalnic.

Metoda normalizacije preprečuje nekonsistentnosti velikosti šarenice. Normalizirana šarenica je vedno enake velikosti in je pretvorjena iz polarne oblike v kartezično. Metoda je bila izdelana na podlagi Daugmanove metode imenovane Daugman's rubber sheet.

Za kodiranje vzorca je bila uporabljena funkcija Gabor filtrov, s katero dobimo unikatni vzorec, katerega je potrebno kvantizirati, da zmanjšamo nabor vrednosti in posledično omogočimo lažje primerjanje vzorcev.

Za primerjanje vzorcev, je uporabljena hammingova razdalja, katero izračunamo na podlagi maske vzorcev.

Sistem je bil narejen z uporabo programskega jezika in okolja MATLAB, za testiranje pa uporabljene sivinske slike iz zbirke CASIA v4.

10.2 Ugotovitve

Z raziskovanjem in implementacijo sistema smo ugotovili, da je segmentacija najbolj pomemben del sistema, saj je to prvi del sistema in napaka pri segmentaciji predstavlja problem za vse nadaljne metode. Prav tako je segmentacija odvisna od zajete slike, če je slika slaba, bo pri segmentaciji težje zaznati pravilno območje, kjer se nahaja šarenica.

Iz testiranja smo ugotovili, da segmentacija najbolje deluje z uporabo Canny edge metode za zaznavanje robov na sliki in s stopnjo zviševanja intenzitete (threshold) med 0.19 in 0.21.

Pomembna ugotovitev je, da uporaba Gabor filtra za kodiranje vzorca ni najboljša rešitev, saj dobimo slabši rezultat za primerjanje vzorcev. Boljša rešitev bi bila uporaba 1D Log Gabor filtera ali pa več dimenzionalnega Log Gabor filtra, kot je predstavljeno v obstoječi Daugmanovi metodi.

Iz rezultatov lahko vidimo, da je ob pravilni izbiri metod, sistem za prepoznavanje oseb na podlagi očesne šarenice lahko zanesljiv in primeren za identifikacijo in avtentikacijo oseb. Prav tako, je časovna kompleksnost presenetljivo nizka, glede na kompleksnost nekaterih delov sistema.

Literatura

[1] J. Daugman. *How iris recognition works.* IEEE Trans. CSVT, vol. 14, no. 1. 2004.

[2] J. Daugman. *High confidence visual recognition of persons by a test of statistical independence.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 15, No. 11, 1993

[3] J. Daugman. *Recognizing persons by their iris patter ns.* Cambridge University Cambridge, UK. 2001.

[4] R P Wildes. *Iris recognition:an emerging biometric technology.* Proc. IEEE, 85(9):1348–1363, 1997.

[5] S V Sheela, P A Vijaya *Iris Recognition Methods - Survey* International Journal of Computer Applications (0975 –8887) Volume 3 –No.5, June 2010

[6] Daugman, John & Downing, C. *Epigenetic Randomness, Complexity and Singularity of Human Iris Patterns.* (2001) Proceedings of the Royal Society: Biological Sciences, 268, 1737-1740.

[7] CASIA v4 *URL:* *http://biometrics.idealtest.org/dbDetailForUser.do?id=4*

[8] Matlab *URL:* *https://www.mathworks.com/*

[9] Tania Johar, Pooja Kaushik *Iris Segmentation and Normalization using Daugman's Rubber Sheet Model* Department of E&C Engineering, Maharishi Markandeshwar Engineering College, MMU, Ambala, Haryana, India-133207

[10] John F. Canny *Finding Edges and Lines in Images.* M.I.T. Artificial Intelligence Lab., Cambridge, Massachusetts, Rep. Al-TR-720, 1983.

[11] Upasana Tiwari, Deepali Kelkar, Abhishek Tiwari *Study of Different IRIS Recognition Methods* International Journal of Computer Technology and Electronics Engineering (IJCTEE) Volume 2, Issue 1

[12] Libor Masek *Recognition of Human Iris Patterns for Biometric Identification* School of Computer Science and Software Engineering, The University of Western Australia, 2003

[13] SANS Institute *Iris Recognition: Closer Than We Think* SANS Institute InfoSec Reading Room Miltiades Leonidou 2002 Assignment Number: 1.4b

[14] MATLAB: Canny edge metoda *URL:* `https://www.mathworks.com/help/images/ref/edge.html`

[15] MATLAB: Circular Hough transform metoda *URL:* `https://www.mathworks.com/help/images/ref/imfindcircles.html`

[16] MATLAB: Linear Hough transform metoda *URL:* `https://www.mathworks.com/help/images/ref/houghlines.html`

[17] Libor Masek, Peter Kovesi. *MATLAB Source Code for a Biometric Identification System Based on Iris Patterns.* The School of Computer Science and Software Engineering, The University of Western Australia. 2003. *URL:* `http://www.peterkovesi.com/studentprojects/libor/sourcecode.html`